# The Crotal SRL System : a Generic Tool Based on Tree-structured CRF

Erwan Moreau, Isabelle Tellier

# The Crotal SRL System : a Generic Tool Based on Tree-structured CRF*

**Erwan Moreau**
LIPN - CNRS UMR 7030 & Univ. Paris 13
Erwan.Moreau@lipn.univ-paris13.fr

**Isabelle Tellier**
LIFO - Univ. Orléans
Isabelle.Tellier@univ-orleans.fr

## Abstract

We present the Crotal system, used in the CoNLL09 Shared Task. It is based on XCRF, a highly configurable CRF library which can take into account hierarchical relations. This system had never been used in such a context thus the performance is average, but we are confident that there is room for progression.

## 1 Introduction

In this paper we present the Crotal Semantic Role Labelling (SRL) system, which has been used in the CoNLL 2009 Shared Task (Hajič et al., 2009)[1]. This system is based on Conditional Random Fields (CRF) (Lafferty et al., 2001; Sutton and McCallum, 2006): our idea is that we can use the provided dependency structure as the skeleton of a graphical model expressing independence asumptions in a CRF model. CRF are a powerful machine learning technique that has been successfully applied to a large number of natural language tasks, mainly to tag sequences. Compared to classification techniques, CRF can easily take into account dependencies among annotations: it is therefore possible to represent tree-like structures in the input of the algorithm. Recently, CRF using tree structures were used in (Finkel et al., 2008) in the case of parsing.

Before participating to this Shared Task, our prototype had only been used to annotate function tags in a French Treebank: these data were drastically smaller, and the task was simpler. Therefore CoNLL 2009 ST is the first time the Crotal System is run for a quite complex task, with so many data as input, and seven different languages (Catalan, Spanish (Taulé et al., 2008), Chinese (Palmer and Xue, 2009), Czech (Hajič et al., 2006), English (Surdeanu et al., 2008), German (Burchardt et al., 2006) and Japanese (Kawahara et al., 2002)). In this context, the performance we obtained seems reasonable: our average F1-measure is 66.49% (evaluation dataset).

One of the advantages we want to emphasise about our system is its genericity: the system does not need a lot of information as input (we mainly use *pos* and *deprel* columns, and the frame sets have not been used), and it was able to achieve satisfying results for the seven different languages using nearly the same parameters (differences were essentially due to the volume of data, since it was sometimes necessary to reduce the processing time). Of course, we hope to improve this prototype thanks to this experience: it may become necessary to lose in genericity in order to gain in performance, but our goal is to maintain as much as possible this advantage.

In section 2 we explain the general architecture for Crotal, then we explain how features are selected in our system in section 3, and finally we detail and discuss the results in section 4.

## 2 The Crotal System Architecture

### 2.1 General principle

The system we propose is based on the public library XCRF (Gilleron et al., 2006; Jousse, 2007), which

*This work has been funded by the French National project ANR-07-MDCO-03 “CRoTAL”.

[1]We have participated in the SRL-only category.

implements CRF model(s) to learn to annotate trees represented by XML documents. Of course, its performance depends on the way it is used, and especially on how *features* are chosen to reliably represent the labeled data. In order to keep the system as generic as possible, features are generated automatically and only a few parameters may vary. The global process has been divided into a sequence of steps, by creating clusters (one for each predicate, except the less frequent ones). Indeed, one expects that the behaviour of the arguments for a given predicate is more regular than for all predicates put together. Moreover, the size of the training set for all seven languages allows such a clustering, and it would even be difficult to process the whole set of predicates due to time and memory limitations. Thus the global process is[2]:

1. Data conversion from CoNLL format to XCRF format:
   - For each sentence containing $n$ predicates, generate $n$ different XML trees[3].
   - The tree is simply built following the dependencies (as provided by the *head* column). Therefore the possible non-projectivity of a tree is ignored, though the order of words is of course prefered whenever possible. An artificial root node is always added (useful for languages where several roots are possible).
   - In each such XML tree, there is only one (marked) predicate, and in the annotated version its arguments (extracted from the corresponding column) and only them are reported in the corresponding nodes.

   Figure 1 shows the labeled XML tree obtained for a (part of) example sentence.

2. Clustering by *lemma*: all dependency trees having the same lemma as predicate are put together if the number of such trees is at least a given threshold (generally 3, also tested with 2 to 5). There is a special cluster for less frequent lemmas[4]. Then, for each cluster, in training mode the process consists of:
   (a) Generation of features for the arguments training step.
   (b) The CRF model for arguments is trained with XCRF.
   (c) Generation of features for the senses training step.
   (d) The CRF model for senses[5] is trained with XCRF.

   In annotation mode, the CRF model for arguments is first applied to the input tree, then the CRF model for senses (if possible, an individual evaluation is also computed).

3. Back conversion from XCRF format to CoNLL format (in annotation mode).

In the framework of this task, features generation is crucial for improving performance. That is why we will mainly focus on that point in the remaining of this paper.

### 2.2 The XCRF Library

XCRF (Gilleron et al., 2006; Jousse, 2007) is a public library which has been applied successfully to *HTML* documents in order to extract information or translate the tree structure into *XML* (Jousse, 2007). More recently we have applied it to annotate function tags in a French Treebank.

In a CRF model, a feature is a function (usually providing a boolean result) whose value depends on the annotations present in a special *clique* of the graph, and on the value of the observed data. In our system, each feature is defined by a pair $(C, T)$, where:

- $C$ is the set of annotations present in a given clique, i.e. a completely connected subgraph of the graphical structure between annotations.

[2] Remark: unless stated otherwise, we will use terms "lemma", "POS tag" "dependency relation" or "head" to refer to the information contained in the corresponding "P-columns" for each word. It is worth noticing that performance would be better using the "real" columns, but we have followed the instructions given by the organizers.

[3] Thus sentences with no predicate are skipped and several trees possibly correspond to the same sentence.

[4] This special cluster is used as a default case. In particular, if an unknown lemma is encoutered during annotation, it will be annotated using the model learned for this default cluster.

[5] Steps 2c and 2d are skipped if the lemma has only one possible sense (or no sense is needed, like in Japanese data and for some Czech predicates).

Several solutions are possible to choose this graph. In most of our experiments, we have chosen a graph where only the node-parent relationship between nodes is taken into account (denoted FT2), as illustrated by Figure 2. XCRF is also able to deal with simple one-node cliques (no dependency between annotation, denoted FT1) and node-parent-sibling relationship (denoted FT3).

- $T = \{t_1, \ldots, t_n\}$ is a (possibly empty) set of boolean tests on the observation (i.e. not depending on the annotations). Each $t_i$ is an atomic test[6]: for example, the test "*pos* attribute for first left sibling is NNS" is satisfied for node 3 in fig. 1. $T$ is the conjunction of all $t_i$.

For example, let us define the following FT2 feature $(C, T)$, that would be true for node 4 in fig. 1: $C$ is $\{apred_{parent} = \text{PRED} \wedge apred_{current} = \text{C-A1}\}$ and $T$ is $\{pos_{child_1} = \text{VB} \wedge deprel_{parent} = \text{VC}\}$.

## 3 Selecting Features

Our goal is somehow to "learn" features from the training set, in the sense that we do not explicitly define them but generate them from the corpus. The main parameters we use for generating a set of features are the following:

- The feature type $n$, with $n \leq 3$. All FT $n'$, with $n' \leq n$, are also considered, because some function tags possibly appear in FT $n$ and not (or more rarely) in FT $n+1$.
- Various kind of accessible information (decomposed through two distinct parameters *information* and *neighbourhood*):
  - Information: form, lemma, POS tags, dependency relation and various secondary attributes (column *features*) are available for all nodes (i.e. word), in every tree extracted from the corpus.
  - Neighbourhood: Given a current node, the "neighbourhood" defines the set of nodes

[6] A test is provided to XCRF as an XPath expression, which will be applied to the current node in the XML tree corresponding to the sentence.

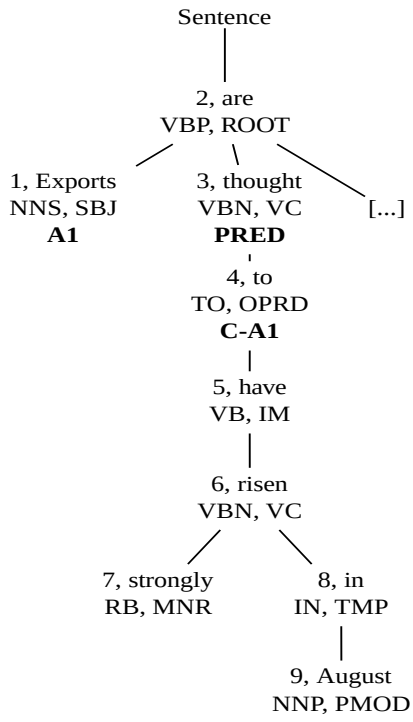


Figure 1: a labeled example for the (part of) sentence "Exports are thought to have risen strongly in August [...]": the nodes are represented with their POS tags, and in bold face the corresponding annotation associated with the predicate "thought" (label PRED was added during preprocessing, see 3.1)

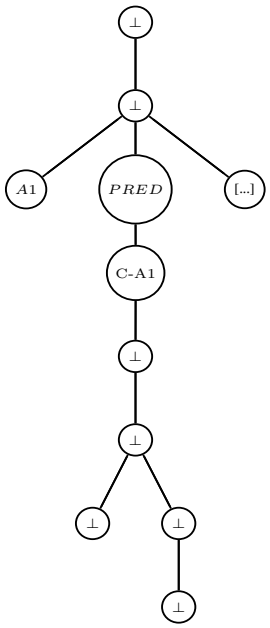


Figure 2: graph for a FT2-CRF for the annotation of the sentence of Figure 1 (where ⊥ means "no annotation")

that will be observed to help deduce its annotation: only this node, or also its parent, possibly its siblings, etc.

- The maximum number of (atomic) tests in the set $T$ for these nodes: combining several tests makes features more precise (conjunction), but also more numerous.

A few other parameters may be added to speed up learning:

- minimum proportion for an argument label which is present in the data to be taken into account,
- minimum proportion for a feature which is present in the data to be included in the model,
- and maximum number of sentences to process by XCRF in the training step.

We try to use as less linguistic knowledge as possible, because we are interested in testing to what extent the model is able to learn such knowledge by itself. Moreover, we observe that using too many features and/or examples as input in XCRF requires a lot of time and memory (sometimes too much), so we have to restrict the selection to the most relevant kind of information in order to get a tractable machinery. This is why we use only POS tags (*pos*) and dependency relations (*deprel*) (as one can see in fig. 1). Finally the process of generating features consists in parsing the training data in the following way: for each encoutered clique, all the possible (combinations of) tests concerning the given neighbourhood are generated, and each of them forms a feature together with the observed clique.

### 3.1 Learning Argument Roles

In our system, the arguments and the sense of a predicate are trained (or annotated) one after the other: the former is always processed before the latter, thus the dependency holds only in the direction from arguments to sense. Therefore the training of arguments only relies on the observed trees (actually only the neighbourhood considered and the arguments cliques). In order to help the learner locate the right arguments, a special label *PRED* is added as "argument" to the node corresponding to the target predicate: by this way cliques can more easily take the tree structure into account in the neighbourhood of the predicate.

After some tests using the development set as test set, we observed that the following parameters were the best suited to build a reliable CRF model (for the arguments) in a reasonable time (and thus used them to learn the final models): the neighbourhood consists in the node itself, its parent and grand-parent, first and second siblings on both sides and first child; the FT2 model performs quite correctly (FT3 has been discarded because it would have taken too much time), and at most two tests are included in a feature.

### 3.2 Learning Predicate Senses

The step of predicting senses can use the arguments that have been predicted in the previous step. In particular, the list of all arguments that have been found is added and may be used as a test in any feature. We did not use at all the frame sets provided with the data: our system is based only on the sentences. This choice is mainly guided by our goal to build a generic system, thus does not need a lot of input information in various formats. The lemma part of the predicate is simply copied from the *lemma* column (this may cause a few errors due to wrong lemmas, as observed in the English data).

The fact that sentences have been classified by lemma makes it convenient to learn/annotate senses: of course lemmas which can not have more than one sense are easily processed. In the general case, we also use XCRF to learn a model to assign senses for each lemma, using the following parameters: there is no need to use another model than FT1, since in each tree there is only one (clearly identified) node to label; a close neighbourhood (parent, first left and right siblings and first child) and only two tests are enough to obtain satisfactory results.

## 4 Results and Discussion

### 4.1 General Results

Due to limited time and resources, we had to relax some time-consuming constraints for some clusters of sentences (concerning mainly the biggest training sets, namely Czech and English): in some cases, the

threshold for a feature to be selected has been increased, resulting in a probably quite lower performance for these models. Ideally we would also have done more tests with all languages to fine-tune parameters. Nevertheless, we have obtained quite satisfying results for such a generic approach: the average F1-measure is 66.49%, ranging from 57.75% (Japanese) to 72.14% (English). These results show that the system is generic enough to work quite correctly with all seven languages[7].

### 4.2 Internal Evaluation

Here we report detailed results obtained in annotating the development set. Since we process the task in two distinct steps, we can evaluate both separately: for the arguments step, the F1-measure ranges from 56.0% (Czech) to 61.8% (German), except for Japanese data where it is only 27%. For the senses step, the F1-measure is generally better: it ranges from 61.5% for the Czech case[8] to 93.3% for Chinese.

It is also interesting to observe the difference between using "real" indicators (i.e. *lemma*, *pos*, *deprel* and *head* columns) versus predicted ones (i.e. *P*-columns): for example, with German data (respectively Catalan data) the F1-measure reaches 73.6% (resp. 70.8%) in the former case, but only 61.8% (resp. 60.6%) in the latter case (for the argument labeling step only).

### 4.3 Impact of Parameters

At first we intended to use the most precise CRF model (namely FT3), but the fact that it generates many more features (thus taking too much time) together with the fact that it does not improve performance a lot made impossible to use it for the whole data. More precisely, it was possible but only by setting restrictive values for other parameters (neighbourhood, thresholds), which would have decreased performance. This is why we had to use FT2 as a compromise, thus making possible to use better values for the other parameters. We have also tested using 3 tests instead of only 2, but it does not improve performance, or not enough to compensate for the huge number of generated features, which requires excessive time and/or memory for XCRF learning step.

One of the most important parameters is the neighbourhood, since it specifies the location (and consequently the amount) of the information taken into account in the features. We have tried different cases for both the argument labeling step and the sense disambiguation step: in the former case, observing children nodes is useless, whereas observing the parent and grand-parent nodes together with two siblings in both left and right handside improves the model. On the contrary, in the senses step observing more than close nodes is useless. These facts are not surprising, since arguments are generally hierarchically lower than predicates in the dependency trees.

We have also studied the problem of finding an optimal threshold for the minimum number of sentences by cluster (all sentences in a given cluster having the same lemma for predicate): if this threshold is too low some clusters will not contain enough examples to build a reliable model, and if it is too high a lot of sentences will fall in the default cluster (for which the model could be less precise). But surprisingly the results did not show any significant difference between using a threshold of 2, 3 or 5: actually individual results differ, but the global performance remains the same.

Finally a word has to be said about "efficiency parameters": the most important one is the minimum proportion for a generated feature to be included in the final set of features for the model. Clearly, the lower this threshold is, the better the performance is. Nevertheless, in the framework of a limited time task, it was necessary to set a value of 0.0005% in most cases, and sometimes a higher value (up to 0.001%) for the big clusters: these values seem low but prevent including a lot of features (and probably sometimes useful ones).

## 5 Problems, Discussion and Future Work

Since there was a time limit and the system was used for the first time for such a task, we had to face

[7] Actually detailed evaluation shows that the system does not deal very well with Japanese, since locating arguments is harder in this language.

[8] Counting only "real senses": it is worth noticing that Czech data were a bit different from the other languages concerning senses, since most predicates do not have senses (not counted here and easy to identify) and the set of possible senses is different for each lemma.

several unexpected problems and solve them quite rapidly. Therefore one may suppose that our system could perform better, provided more tests are done to fine-tune parameters, especially to optimize the balance between efficiency and performance. Indeed, there is a balance to find between the amount of information (number of features and/or examples) and the time taken by XCRF to process the training step. Generally speaking, performance increases with the amount of information, but practically XCRF can not handle a huge number of features and/or examples in a reasonable time. This is why selecting the "right" features as soon as possible is so important.

Among various possible ways to improve the system, we should benefit from the fact that CRF do not need a lot of examples as input to learn quite correctly. Informally, the XCRF library seems to have some kind of "optimal point": before this point the model learned could be better, but beyond this point time and/or memory are excessive. Thus one can try for example to apply an iterative process using a sufficiently low number of features at each step, to select the more useful ones depending on the weight XCRF assigns to them.

Since the Crotal system obtained reasonable results in this "non ideal" context, we are quite confident in the fact that it can be significantly improved. The CoNLL 09 Shared Task has been a good opportunity to validate our approach with a non trivial problem. Even if the performance is not excellent, several important points are satisfying: this experience shows that the system is able to handle such a task, and that it is generic enough to deal with very different languages.

## References


Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of ACL-08:HLT*, pages 959–967, Columbus, Ohio. Association for Computational Linguistics.

Rémi Gilleron, Florent Jousse, Isabelle Tellier, and Marc Tommasi. 2006. Conditional random fields for xml trees. In *Proceeding of ECML workshop on Mining and Learning in Graphs*.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. *Prague Dependency Treebank 2.0*. Linguistic Data Consortium, Philadelphia, Pennsylvania, USA. URL: http://ldc.upenn.edu. Cat. No. LDC2006T01, ISBN 1-58563-370-4.

Florent Jousse. 2007. *Transformations d'Arbres XML avec des Modèles Probabilistes pour l'Annotation*. Ph.D. thesis, Université Charles de Gaulle - Lille 3, October.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML'01: Proceedings of the 18th International Conf. on Machine Learning*, pages 282–289.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Charles Sutton and Andrew McCallum. 2006. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.